Supplement 2:

再谈一次 AI 的逻辑结构

YKY 甄景贤

July 31, 2020

Table of contents

背景：利用 invariance 加速学习

- 以前曾经说过，机器视觉 的成功，有赖於 将 视觉的几何结构 impose 在深度神经网络上
- 这 深度神经网络 原本是 “free” 的，但加了限制之后，权重空间 变小了（例如维数降低），所以学习加速了
- 所谓 symmetry 的意义，简单例子：「如果知道左边等於右边，那就只需计算一次」
- 换句话说，数学家喜欢对称性，是因为它经常可以简化计算
- 同理，我们想将 逻辑结构 的对称性 impose 到神经网络
- 实际上，可能只需要逻辑上的交换律，就可以达到 强人工智能，正如 机器视觉的成功，在於引入了 CNN 的 convolution 结构，后者只是 视觉不变性 的其中一个最显著的 invariant
- 现代逻辑理论非常漂亮，我花了十多年时间才弄懂，我希望将这套 逻辑-学习 理论简单讲解一下，也算功德完满了

逻辑的 几何 与 拓扑 结构

- 在经典时代，逻辑的 代数形式 可以用 Boolean algebra 表述，然而这方法只适用於 命题逻辑
- Boolean algebra 是中学生熟悉的，类似 Venn diagram 的结构
- 这种结构和 拓樸学 的 open sets 结构一样，所以 命题逻辑 也可以看成是一种 topology
- 然而 predicate logic 的结构更复杂，直到最近才有比较完善的表述
- 现代逻辑结构和 type theory 有深刻的关系，此即 Curry-Howard isomorphism
- 现代逻辑也涉及 topos theory，那是一种由 algebraic geometry 引入的结构

Curry-Howard isomorphism

- Curry-Howard isomorphism 是现代逻辑的 核心思想，但我初时没有留意，以致很多东西 看不懂，明白之后 豁然开朗
- 它讲的是 逻辑证明 与 计算 之间的深刻关系：

$$\begin{aligned} \text{逻辑} &\Leftrightarrow \text{type theory} \\ \text{逻辑证明} &\Leftrightarrow \text{programs} \end{aligned} \tag{1}$$

- 先看 $\Rightarrow$：逻辑是一些 符号 / 形式上 的推导，所以 逻辑证明 (derivations) 必然对应於某种 计算 (computation)，这很容易理解
- 再看 $\Leftarrow$：当人们企图定义 普遍的计算模式 (models of computation) 时，发觉 总是对应於 某些逻辑；这一点比较神秘，我也不完全了解

Type theory

- 首先了解一下 type theory, 它起源於 Bertrand Russell 为了避开 数理逻辑上的 悖论 (paradox) 的尝试
- 后来发现 type thoery 用来定义 programs 很有用
- 大家都知道 Lisp 语言没有 types，它是一种 untyped λ-calculus
- 在 Lisp 之上引入 type system，衍生成 ML, Caml, OCaml, Haskell, 等 一系列的语言
- 每一个 program 属於某个 type，例如 $\mathrm{length}()$ 函数：

$$\mathrm{length} : \mathsf{String} \to \mathsf{Integer} \tag{2}$$

它输入一个 字串，输出一个 整数（字串的长度）

Curry-Howard isomorphism 的一些细节

- 例如 我们定义一个函数 f, 由 A 类 映射到 B 类:

$$f : A \to B \tag{3}$$

- 这对应於 逻辑上「A 蕴涵 B」的关系:

$$A \Rightarrow B \tag{4}$$

- 逻辑上，每个命题 A 都有一个 proof object 或 witness, 记作 $\square : A$
- 而函数 f 就是将 $a : A$ （A 的证明）映射到 $b : B$ （B 的证明）:

$$f : a \mapsto b \tag{5}$$

- 类似地，有函数可以将 两个分开的命题 A 和 B 的证明 映射到 $A \wedge B$ 的证明，这里不赘述了
- 简言之，可以建立 命题逻辑 的 $\Rightarrow, \wedge, \vee, \neg$ 对应到一些函数上，这些函数用 typed λ-calculus 定义
- 但与 λ-calculus 对应的逻辑 没有 排中律，它是 intuitionistic logic. 这种逻辑 符合 数学上的 构造主义 (constructivism)

谓词逻辑 (predicate logic)

- 刚才说明了，type theory 对应於 命题逻辑
- 如果要处理 predicates，需要一些 特殊的 types
- Predicate 就是有「洞」的命题，填入某些 objects 之后变成真正的命题
- 换句话说 predicate P 是一个函数 $P : X \to \mathcal{U}$，对每个 x 产生 $P(x)$，这 $P(x)$ 也是一个 type, 而 $\mathcal{U}$ 是所有 types 的 universe
- 这样会产生一些不是很像「命题」的 types，例如 自然数的类 $\mathbb{N}$, 似乎不是命题；关於这点我暂时仍不太清楚
- 重点是：type theory 很自然地 同构於 categories，with objects $=$ types and morphisms $=$ function types $(A \to B)$
- 而在 categories 中，predicate 的结构可以用 fibration 描述，记作 $\begin{array}{c}\mathbf{Pred}\\ \downarrow \\ \mathbf{Set}\end{array}$
- Fibrations 符合某种 universal property，是由 代数几何 / 拓扑 借来的概念

Curry-Howard isomorphism 的意义

- 有些逻辑学家 察觉到 类型论 的 $A \to B$ 和逻辑中 $A \Rightarrow B$ 是一模一样的
- 这个关系的 发现者 至少包括: Brouwer-Heyting-Kolmogorov-Schönfinkel-Curry-Meredith-Kleene-Feys-Gödel-Läuchli-Kreisel-Tait-Lawvere-Howard-de Bruijn-Scott-Martin-Löf-Girard-Reynolds-Stenlund-Constable-Coquand-Huet-....
- 这关系的深刻之处，在於把 符号逻辑上的 proofs 和 程式语言 的 programs 划上等号，前者是 符号 / 静态的，后者是 程序 / 动态的
- 每个 proof 就是一个 program，它输入一些 arguments，输出 关於那些 arguments 的证明
- 例如：「所有人都会死」是一个 program，它输入「苏格拉底」，输出「苏格拉底会死」
- 我发现 这个对应 也可以应用到 深度学习：神经网络 也是一種 函數 / mapping，它将 逻辑前提 map 到结论

Topos theory and fibrations

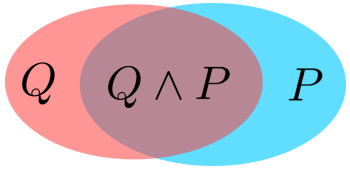
- Predicate logic （谓词逻辑）和 命题逻辑 之间的差异在於 fibration 结构
- Fibration 通常用 $\begin{matrix}\mathbb{E}\\ \downarrow\pi\\ \mathbb{B}\end{matrix}$ 表示，$\mathbb{B}$ = base space, $\mathbb{E}$ = ètalè space, π = projection
- 例如 base space 是一个 base set, ètale space 是这个集合上的 predicates
- 由於 Curry-Howard 对应, type = propositions，在 base 空间上只有 命题逻辑
- 例如 $\mathbb{B}$ 空间的一个 type 是 Human，$\mathbb{E}$ 空间的一个谓词是 Mortal
- 於是有以下这个 type inference rule:

$$i : \mathrm{Human} \vdash \mathrm{Mortal}(i) : \mathrm{Prop} \tag{6}$$

意思是说，如果 i 属於 Human 类型，则 $\mathrm{Mortal}(i)$ 属於 Prop 类型

Logic's topology and geometry

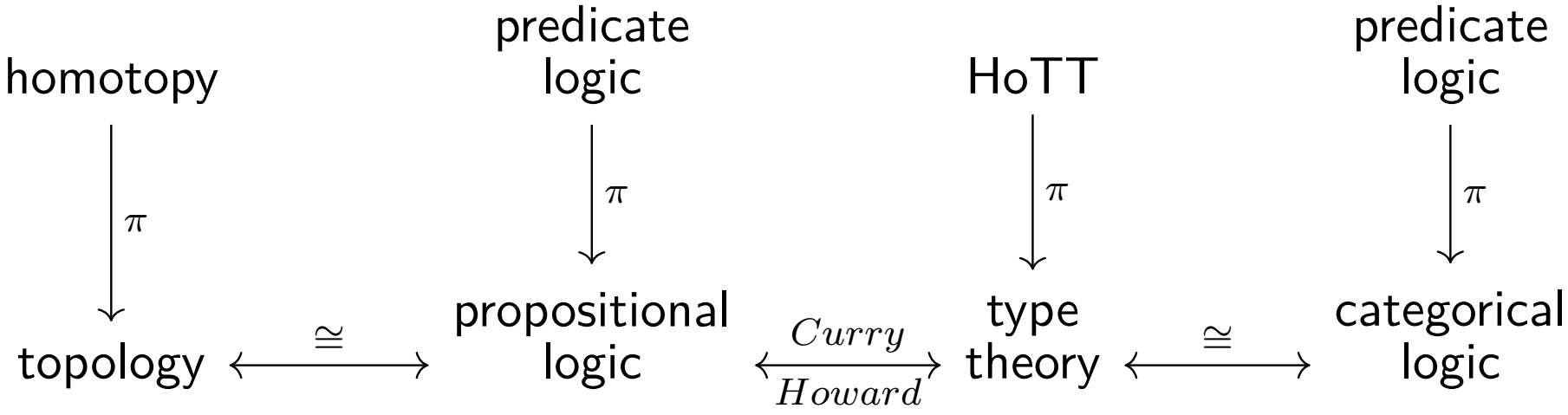
- 总结一下，大家 中 / 小学 时期 都熟悉 Venn diagrams，其实 命题逻辑 具有 拓扑的 open sets 结构：



(7)

而 谓词 (predicates) 是在 base set 上面的 纤维化 (fibration)，记作 $\begin{matrix}\mathbb{E}\\ \downarrow\pi\\ \mathbb{B}\end{matrix}$

- 於是有以下的关系：(π = fibration)



(8)

- HoTT = homotopy type theory, 是 Voevodsky 等人提出的 非常新的理论

HoTT (homotopy type theory)

- 2017 年，提出 HoTT 的 Voevodsky 在 51 岁 英年早逝

HoTT 理论我不太了解，也未知它对 AGI 有没有用，暂时留下这个 remark：

- HoTT 的作用似乎是对 types 的空间作抽象的描述，例如将 types 空间分成 levels:
  - **Level 0:** up to homotopy equivalence there is just one contractible space that we call "point" and denote pt
  - **Level 1:** up to homotopy equivalence there are 2 spaces at this level: the **empty space** $\varnothing$ and the **point** pt. We call them **truth values**. We also refer to types of this level as **properties** and **propositions**. Propositional logic lives at h-level 1
  - **Level 2:** Types of this level are characterized by the property that their path spaces are empty or contractible. So such types are disjoint unions of contractible components (points), or in other words **sets** of points. This will be our working notion of sets available in this framework.
  - **Level 3:** Types of this level are characterized by the property that their path spaces are sets (up to homotopy equivalence). These are obviously (ordinary flat) **groupoids** (with path spaces hom-sets)
  - **Level 4** ...$n+2$**:** Here we get 2-groupoids to n-groupoids

逻辑 与 AI 之间的关系

- 那既然 AI 基於 逻辑，而逻辑的结构 如上所述，则 AI 与逻辑之间 必然存在 精确 (precise) 的联系
- BERT 似乎是在执行 句子之间的变换，而这些句子是 word embedding 的 concatenation，例如：

$$\text{苏格拉底} \cdot \text{是} \cdot \text{人} \xrightarrow{BERT} \text{苏格拉底} \cdot \text{会} \cdot \text{死} \tag{10}$$

这个做法看似很「粗暴」，其实它和逻辑式子的作用一样：

$$\forall x.\ \mathsf{Human}(x) \to \mathsf{Mortal}(x) \tag{11}$$

而这式子，根据 Curry-Howard 对应，就是上面的函数映射!

- 这些映射的对象，是形如 "$a \in A$" 这样的物体，它们有 predicates 的几何 / 拓扑结构，但我们现时未做到这一步，暂时只引入了 commutativity
- 可以说，逻辑的 几何结构 是「永恒」的，它可以指示 AGI 的 长远发展

对 逻辑主义 的质疑

- 很多人怀疑：人脑真的用 逻辑 思考吗？
- 其实我们每句表达的 语言，都是逻辑形式的 (logical form)
- 直觉认为，人脑 构造一些 models，再从 model 中「读出」一些结论
- 例如给定一个描述：「已婚妇人出轨，用刀刺死丈夫」

(12)

- 那么 妻子穿著什么衣服？衣服什么颜色？这些都是 臆想 出来的细节，是不正确的
- 这个 model 可以有哪些细节？答案是：任何细节都不可以有，除非是 逻辑上蕴含的，或被逻辑 约束
- Model 本身可以是一些 抽象的逻辑命题 构成的，这也合理；反而，一个有很多感官细节的 model 并不合理
- 其实人脑可能比我们想像中更接近逻辑化的结构

「神经」知识表示

为什么要研究 神经知识表示?

- 从经典 logic-based AI 的传统，一直在使用「符号」的知识表示法
- 符号逻辑 很容易转换成 抽象代数 / 范畴论 形式（它们是同一个大家庭的「近亲」）
- 然而 或许存在 截然不同 的知识表示法? 但我们很难想像它 长什么样子
- 人脑的「神经」知识表示，可以作为参考，然后再研究它和逻辑表示之间的 correspondence

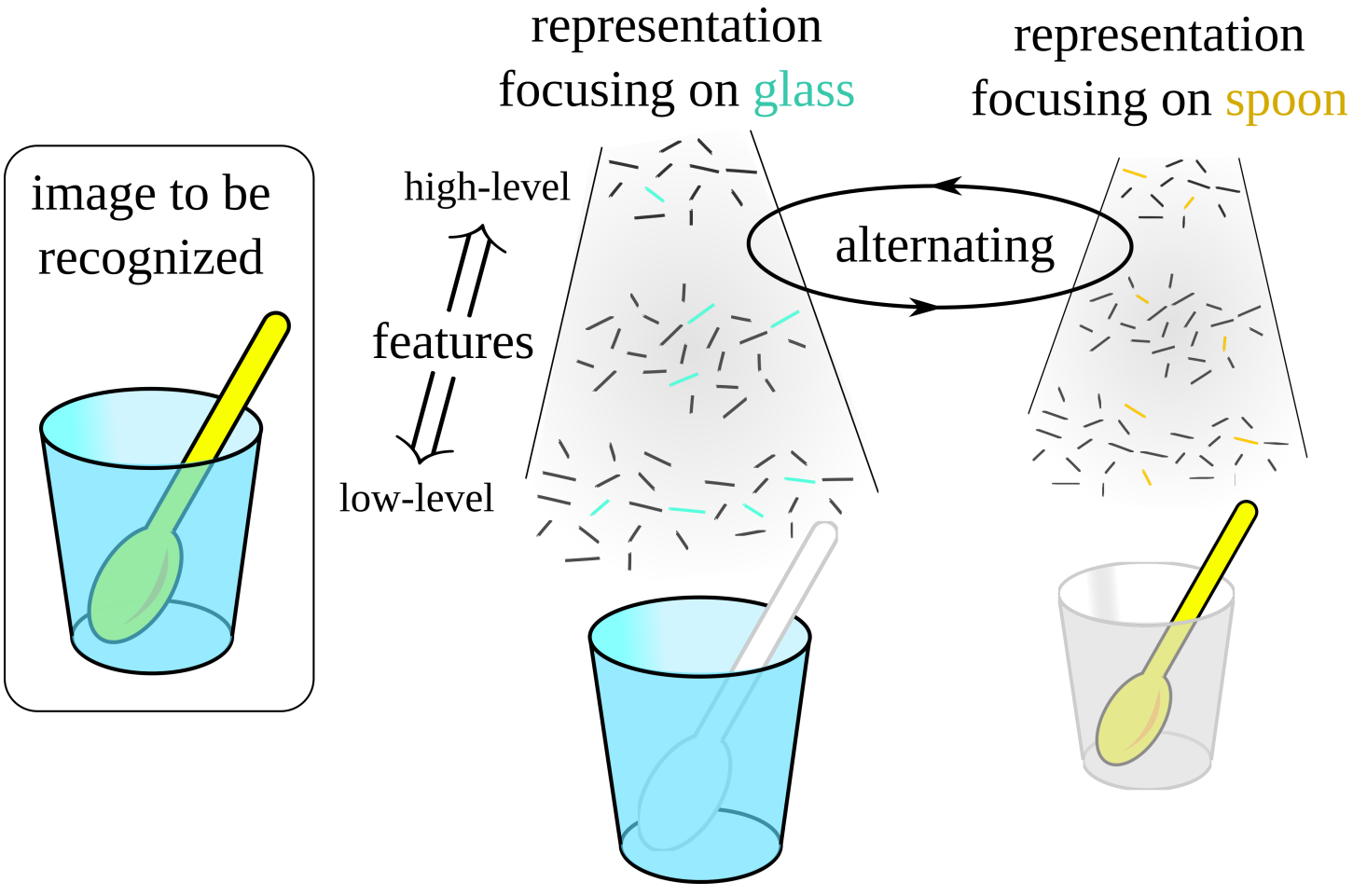
神经知识表示 的特点:

- distributive（分布性）
- model-based (vs rule-based)
- *in situ*（固定性）— 例如辨认「猫」的时候，大脑中 相应的神经元被 激活，但这些 神经元 不能移动，所以「猫」的表示 也不可移动

问题是：如果要辨认「白猫追黑猫」，「猫」的表示是固定的，则这两个「猫」表示 如何共存於神经网络中?
答案很可能是：两个「猫」交替地 出现在 时间上

神经 特征簇 (feature clusters)

例如，以「匙羹在杯中」作例子：

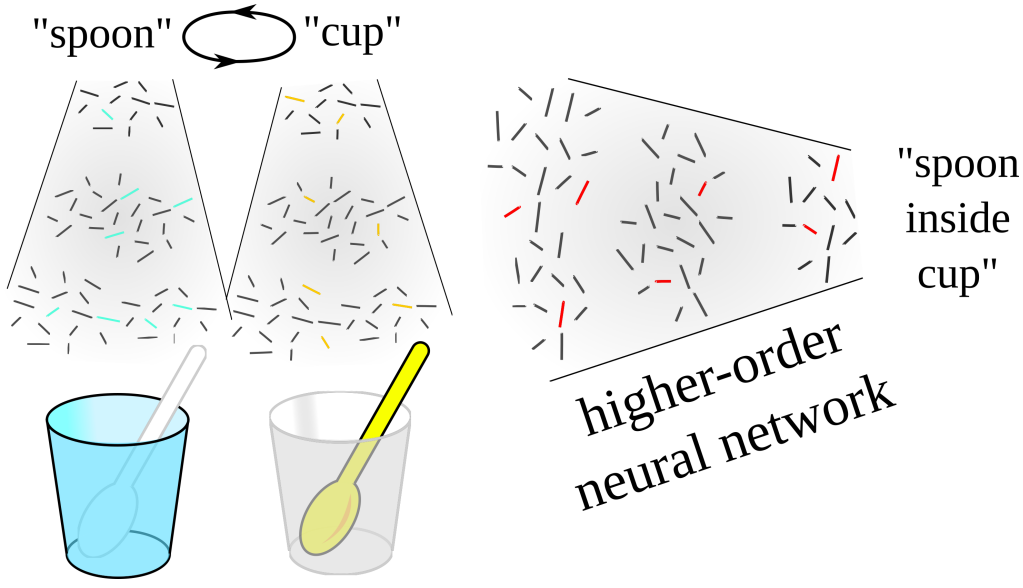


(13)

每个 复杂物体 由一个 feature cluster 辨认。多个「特征簇」在时间上交替出现，可以看成是一种 composition，例如 $A \cdot B$ 或 $A \circ B$.

高阶 特徵

- 一串 特征簇 的时间序列，例如 $A \cdot B$，可以被 更高阶 的神经网络 用作输入。高阶辨认 的结果是一些关系 (relations)，例如「匙羹在杯内」



(14)

- 这似乎是一个 特征空间 $\times$ 时间 的映射 $f : X \times T \to Y$
- 关於这部分其实我仍未肯定，或许有其他方法

References

多谢收看 ☺️

[1] Abramsky and Tzevelekos. "Introduction to categories and categorical logic". In: *New structures for physics*. Ed. by Coecke. 2011. Chap. 1.

[2] Awodey. *Category theory*. 2006.

[3] Caramello. *Theories, sites, toposes – relating and studying mathematical theories through topos-theoretic 'bridges'*. 2018.

[4] Goldblatt. *Topoi – the categorical analysis of logic*. 1984, 2006.

[5] Bart Jacobs. *Categorical logic and type theory*. Elsevier, 1999.

[6] Lawvere and Rosebrugh. *Sets for mathematics*. Cambridge, 2003.

[7] Lawvere and Schanuel. *Conceptual mathematics*. Cambridge, 1997, 2009.

[8] Saunders MacLane and Ieke Moerdijk. *Sheaves in geometry and logic – a first introduction to topos theory*. Springer, 1992.

[9] Andrew Pitts. *Notes on categorical logic*. 1989, 1991.

[10] Andrei Rodin. *Axiomatic method and category theory*. 2014.

[11] Streicher. *Domain-theoretical foundations of functional programming*. World Scientific, 2006.

[12] Vickers. *Topology via logic*. 1989.